Attorney Docket No.: 8971-0005

UNITED STATES PATENT APPLICATION

FOR

SYSTEMS AND METHODS FOR ADDRESS SPACING IN A FIREWALL CLUSTER

BY

STEVEN Y. ZHOU

BACKGROUND OF THE INVENTION

I. Field of the Invention

[001] The present invention generally relates to communication systems and, in particular, to systems and methods for determining addresses.

II. Background and Material Information

[002] A firewall protects the resources of a private network from users (or computers) of other, untrusted networks. The firewall may consist of a set of programs that regulate the flow of traffic into and out of a network. The set of programs may include rules that represent a security policy for the firewall. For example, an enterprise with an intranet that allows its workers access to the wider Internet may install a firewall to control access to the enterprise's own private data resources and to control what outside Internet resources the enterprise's users may access. A firewall may also include (or work with) a proxy server that makes network requests (i.e., to establish connections and exchange packets) on behalf of intranet users. In some situations, a firewall may be a specially designated computer separate from the rest of the computers on a network, such that no incoming packets (or connections) from or to an untrusted network can directly access computers on the enterprise's intranet.

[003] Since a firewall essentially serves to control the flow of packets between a trusted network (e.g., an enterprise or corporate intranet) and an untrusted network (e.g., the Internet), the firewall may be burdened with a large amount of network traffic. To address problems that flow from significant network

traffic some have used a firewall cluster. The firewall cluster includes a plurality of firewalls (also referred to as firewall nodes). By using multiple firewall nodes, the firewall cluster can handle more traffic as compared to a single firewall node. However, the introduction of firewall clusters presents unique problems, such as coordinating the activities of all of the firewall nodes in the cluster.

SUMMARY OF THE INVENTION

[004] Accordingly, the present invention is directed to systems and methods for determining outgoing source addresses in a set of nodes, and, more particularly, for determining such addresses in a firewall cluster.

[005] In one embodiment consistent with the present invention, there are provided systems and methods for receiving, at a first node, a first packet. Moreover, systems and methods are provided for determining as a function of a multidimensional space for representing addresses processed by a set of data processors, a first address for the first packet. Furthermore, systems and methods are provided for forwarding the first packet based on the determined first address.

[006] It is to be understood that both the foregoing general description and the following detailed description are exemplary and explanatory only and are not restrictive of the invention, as described. Further features and/or variations may be provided in addition to those set forth herein. For example, the present invention may be directed to various combinations and subcombinations of the disclosed features and/or combinations and subcombinations of several further features disclosed below in the detailed description.

BRIEF DESCRIPTION OF THE DRAWINGS

[007] The accompanying drawings, which are incorporated in and constitute a part of this specification, illustrate various embodiments and aspects of the present invention and, together with the description, explain the principles of the invention. In the drawings:

[008] FIG. 1 illustrates an exemplary system environment in accordance with systems and methods consistent with the present invention;

[009] FIG. 2 is an exemplary data processor in accordance with systems and methods consistent with the present invention;

[010] FIG. 3 is an exemplary flowchart depicting steps for determining a source address of an outgoing connection associated with a firewall in accordance with systems and methods consistent with the present invention;

[011] FIG. 4A is an exemplary packet received by a firewall cluster in accordance with systems and methods consistent with the present invention;

[012] FIG. 4B is an exemplary packet sent by a firewall cluster in accordance with systems and methods consistent with the present invention;

[013] FIG. 5 illustrates a three-tuple space in accordance with systems and methods consistent with the present invention;

[014] FIG. 6 is another exemplary flowchart depicting steps for determining an N-tuple address in accordance with systems and methods consistent with the present invention;

[015] FIG. 7 is a functional block diagram for determining a quadrant identifier based on a hash function in accordance with systems and methods consistent with the present invention; and

[016] FIG. 8 depicts three quadrant identifiers and respective firewall nodes in accordance with systems and methods consistent with the present invention.

DETAILED DESCRIPTION

[017] Reference will now be made in detail to the invention, examples of which are illustrated in the accompanying drawings. Wherever possible, the same reference numbers will be used throughout the drawings to refer to the same or like parts.

[018] FIG. 1 includes a set of clients 1010-1030, a first network 1205, a firewall cluster 1500, a second network 1210, and a set of servers 1040-1060. Firewall cluster 1500 may further include one or more firewall nodes 1510-1530.

[019] In one embodiment consistent with the present invention, a firewall node (e.g., a firewall node of a cluster) may receive a packet and before sending the packet to its destination, the firewall node determines a source address for the packet, such that the address does not conflict with source addresses assigned by the other firewall nodes.

[020] In one embodiment, the firewall node makes the address based on a multidimensional space assigned to the firewall node making the determination, with the address being representative of a Transport Control Protocol (TCP) connection. The multidimensional space is also referred to as an N-tuple space. The term N-tuple space means multiple in degree N. For example, a 2-tuple space represents a two-dimensional space (or region), a 3-tuple represents a three-dimensional space, and so forth. For example, a firewall node may be assigned a region in N-tuple space that is separate from and does not conflict with any other region assigned to

other firewall nodes of the firewall cluster. As such, when the firewall node determines an address based on its assigned N-tuple space, the resulting address does not conflict with an address assigned by one of the other firewall nodes.

[021] An address of a packet may represent any value that may be used to identify the packet, its source, and its destination. In the case of a TCP packet, the connection may be defined by one or more of the following: a source address, a source port address, a destination address, a destination port address (also known as a port number), and a protocol byte. If five values are used as an address to identify the connection (e.g., a source addresses, a source port address, a destination address, a destination port address, and a protocol byte), these five values are a 5-tuple address that represent a point in 5-tuple space. As such, when a firewall node determines a 5-tuple address based on its assigned region in 5-tuple space, the 5-tuple address cannot conflict with other addresses determined based on other regions in 5-tuple space, which are assigned to the other firewall nodes 1520-1530 of cluster 1500. A 5-tuple space cannot be readily visualized; FIG. 5 (described below) depicts a 3-tuple space.

[022] Clients 1010-1030 may function to establish connections by sending and/or receiving packets through first network 1205 to firewall cluster 1500. Each of the clients may be embodied in the form of a data processing system or computer, as described in greater detail below with respect to FIG. 2.

[023] First network 1205 may function as a communication medium and may include, alone or in any suitable combination, a telephony-based network, a local area network (LAN), a wide area network (WAN), a dedicated intranet, the Internet,

a wireless network, or a bus. Further, any suitable combination of wired and/or wireless components and systems may be incorporated into the communication medium. In one embodiment, first network 1205 may be configured as an intranet, such as a corporate intranet, where security with respect to the public Internet is a concern.

[024] Firewall cluster 1500 may include one or more firewall nodes 1510-1530. Each firewall node may include hardware and/or software that functions to protect first network 1205 and/or client processors 1010-1030. Moreover, the structure of each firewall node may be a data processor (e.g., a computer) and/or a communication device (e.g., a router). Firewall cluster 1500 may also include an arbitrator (not shown) that selects which one of the firewall nodes 1510-1530 will process a packet and its corresponding connection from a client processor. The arbitrator may be embodied as a data processor separate from each of the firewall nodes or, alternatively, may be embodied within one or more of the firewall nodes. The firewall node selected by the arbitrator may then receive the packet and serve to proxy a packet connection to servers 1040-1060 through second network 1210.

[025] Second network 1210 may function as a communication medium and may include, alone or in any suitable combination, a telephony-based network, a local area network (LAN), a wide area network (WAN), a dedicated intranet, the Internet, a wireless network, or a bus. Further, any suitable combination of wired and/or wireless components and systems may be incorporated into the

communication medium. In one embodiment, second network 1210 may be configured as the Internet.

[026] Servers 1040-1060 may function to provide information, such as files, web pages, and other services to clients 1010-1030. Each of the servers may be embodied in the form of a data processing unit (described in greater detail below with respect to FIG. 2). One of ordinary skill in the art would recognize that at times a server may function as client and that a client may function as server.

[027] The data processing unit included in each of the clients 1010-1030, firewall nodes 1510-1530, and servers 1040-1060 may be implemented as a computer 2000, such as the one depicted in general block diagram form at FIG. 2. Computer 2000 may include an input module 2050, a processor 2200, a storage module 2500, and/or an output module 2300.

[028] The output module 2300 may include one or more input/output (I/O) devices including a display 2350, a printer 2360, and a network interface 2380. Network interface 2380 enables computer 2000 to communicate through a network, such as network 1205 and network 1210. For example, network interface 2380 may be embodied as an Ethernet network interface card or a wireless LAN interface card, such as cards compatible with the IEEE 802.11 series of standards.

[029] Input module 2050 of Fig. 2 may be implemented with a variety of I/O devices to receive a user's input and/or provide the input to processing unit 2200. Some of these devices may include, for example, a keyboard, a mouse, an input storage device, a network interface card, and a modem.

[030] Processing unit 2200 may include, for example, one or more of the following: a central processing unit, a co-processor, memory, registers, and other processing devices and systems as appropriate. Although Fig. 2 illustrates only a single processor unit 2200, computer 2000 may alternatively include a set of processing units.

[031] Storage module 2500 may be embodied with a variety of components or subsystems capable of providing storage including, for example, a hard drive, an optical drive, a general-purpose storage device, a removable storage device, and/or memory. Further, although storage module 2500 is illustrated in Fig. 2 as being separate or independent from data processing unit 2200, storage module 2500 and data processing unit 2200 may be implemented as part of a single platform or system.

[032] Although data processing unit 2200 is generally described in terms of computer 2000, data processing unit 2200 may also be incorporated into any other data processing or communication device including, for example, a router, a gateway, a bridge, a firewall, a network security system, a wireless (or portable) device, and/or a network management system.

[033] FIG. 3 is an exemplary flowchart depicting steps for determining a 5-tuple address for a packet based on a 5-tuple space consistent with an embodiment of the present invention. Referring to FIGs. 1 and 3, client 1010 may attempt to make a connection to server 1050 by sending an Internet Protocol (IP) packet to destination server 1050 through first network 1205. Firewall cluster 1500 (or an arbitrator therein) may receive the IP packet from client 1010 (being on the

routed path from 1010 to 1050) and select which of the firewall nodes 1510-1530 will further process the packet and any corresponding Transport Control Protocol (TCP) connection (steps 3100-3200). For example, firewall cluster 1500 may designate firewall node 1510 to receive the packet and serve as a proxy firewall—terminating thus the TCP connection from client 1010 (step 3300). Firewall cluster 1500 may then read a 5-tuple address associated with the packet and analyze the 5-tuple address based on a 5-tuple space, such that the determined 5-tuple address does not conflict with any other 5-tuple address used by the other firewall nodes of the firewall cluster (steps 3350-3400). As such, when firewall node 1510 opens a new connection to server 1050 using the determined 5-tuple address, firewall node 1510 sends the packet using the determined 5-tuple address, without any addressing conflicts (steps 3500-3600) with the other nodes 1520-1530.

[034] Before providing a detailed description of steps 3100-3600, a description of the 5-tuple address is now provided with reference to FIGs. 4A and 4B. FIG. 4A shows an IP packet (e.g., the packet received by firewall node 1510 in step 3100) with a 4-byte source address 4010, a 2-byte source port address 4020, a 4-byte destination address 4030, a 2-byte destination port address 4040, a 1-byte protocol 4050, and a variable number of data bytes 4060. The term 5-tuple address refers to the five address values of the packet, namely, source address 4010, source port 4020, destination address 4030, destination port 4040, and protocol 4050. Source address 4010 describes the source of a packet, for example, the sender, within the network. The source port address 4020 describes a logical connection at the source. The destination address 4030 describes the destination of a packet, for

example, the receiver, within the network. The destination port 4040 describes a logical connection (e.g., port address 80 represents the well-known port for hyper text transfer protocol (http) connections) at the destination. Lastly, the protocol describes the protocol associated with the packet (e.g., TCP, UDP, etc.). Table 1 below lists exemplary values for a 5-tuple address. Referring to Table 1, source address 4010 may be represented as an IP address as depicted in Table 1.

TABLE 1 EXEMPLARY 5-TUPLE VALUES

SOURCE ADDR 4010	SOURCE PORT 4020	DESTINATION ADDR 4030	DESTINATION PORT 4040	PROTOCOL 4050
192.168.1.1	4096	200.1.9.1	80	5

[035] For security reasons, some firewalls replace one or more of the values of the 5-tuple address when sending packets, such as replacing source address 4010 with a firewall address associated with firewall cluster 1500, as depicted in FIG. 4B. Moreover, the source port address 4020 may also be replaced with a value not yet assigned locally (e.g. a value between 1024 and 65535). When this is the case, however, there is a likelihood that two firewall nodes will use the same outgoing 5-tuple address on two different packets corresponding to two different connections. For example, when two client processors 1010-1020 send packets to access the same web site (e.g., server), the clients 1010-1020 may send respective packets to the same website. Each of the packets may have the same website destination address 4030, website destination port 4040, and website protocol 4050 (e.g., http). Firewall node 1510 may then receive the respective packet from client 1010, replace the packet source address with the firewall cluster's address, for

example, 10.10.1.10, and replace the packet source port address (or number) with a value, for example, 4096. Meanwhile, firewall node 1520 may receive the respective packet from client 1020, replace the packet source address with the address of 10.10.1.10, and replace the packet source port address with a value of 4096. Even though the source port numbers are independently assigned by the firewall nodes, this example demonstrates that there is a likelihood (or probability) that packets from different clients 1010-1020 will have the same 5-tuple address, i.e., the same source address 4110 of 10.10.1.10, the same source port 4120 of 4096, the same destination address 4130 of the website, the same destination port 4140 of the website, and the same protocol 4150 of http. The same 5-tuple addresses thus result in a conflict, since two different packets share the same 5-tuple address. As such, packets from two different clients can no longer be distinguished—resulting in a lost or denied connection.

[036] To eliminate such conflicts, a firewall node may determine a 5-tuple address for a received packet that avoids any addressing conflicts with other firewall nodes of the firewall cluster. In particular, a firewall node (e.g., firewall node 1510) may determine a 5-tuple address for an outbound packet (or connection), such that the 5-tuple address is based on an address region assigned to firewall node 1510. For example, each firewall node may be assigned its own region of addresses in N-tuple space that does not conflict with another region assigned to any other firewall node of the firewall cluster. As such, a packet sent by firewall node 1510 (depicted in FIG. 4B) may only use a 5-tuple address from the address region assigned to firewall node 1510.

[037] Since it is difficult to draw a five dimensional space, FIG. 5 depicts an exemplary 3-tuple space including source address, source port, and destination address. Referring to FIG. 5, each firewall node is assigned its own region of addresses in 3-tuple space. Firewall node zero 1530 is assigned region 5100, firewall node one 1520 is assigned region 5200, and firewall node two 1510 is assigned region 5300. For example, firewall node 1530 may determine a 3-tuple address having a value of X, Y, Z, with X representative of a source address, Y representative of a source port, and Z representative of a destination address. As can be seen by FIG. 5, the 3-tuple address X, Y, Z (labeled 5120) does not conflict with the 3-tuple address used by other firewall nodes. Moreover, the use of a N-tuple address determined based on an N-tuple space may provide more flexibility when compared to merely pre-assigning source port numbers to each firewall node to avoid address conflicts. Although the use of a 3-tuple addresses and 5-tuple addresses are described herein, any dimension of N-tuple addresses may be used instead.

[038] Referring again to FIGs. 1 and 3, to receive a packet from client 1010 (or network interface 2380 therein) (step 3100), firewall cluster 1500 may listen for a packet with its address, such as a Media Access Control (MAC) address specifying a physical interface associated with the firewall cluster. Alternatively, firewall cluster 1500 (on the routed path to network 1210) may receive any packets with IP addresses not within its (sub)network, namely first network 1205, since such packets may traverse firewall cluster 1500 to reach second network 1210 and servers 1040-1060.

[039] To designate (or select) which of the firewall nodes 1510-1530 will further process the packet and any corresponding TCP connection, firewall cluster 1500 (or arbitrator therein) may randomly assign a packet to one of the firewall nodes 1510-1530. Subsequent packets from the same connection may be sent to the same firewall node as the arbitrator, when the arbitrator uses a symmetric routing algorithm, which is known and commercially available. Alternatively, firewall cluster 1500 may assign the packet to a firewall with the least amount of load (or traffic). One of ordinary skill in the art would recognize that any other way of arbitrating may be used instead. Moreover, such arbitrators are known and commercially available.

[040] As noted above, the selected firewall node (e.g., firewall node 1510) may act as a proxy firewall. When this is the case, firewall node 1510 may terminate the TCP connection associated with the received packet from client 1010 (step 3300). Moreover, firewall node 1510 may use one or more rules (stored in storage module 2500) to determine whether to forward the packet to its destination. A rule may regulate the flow of packets based on packet content, such as preventing a packet to be delivered to a predetermined destination address or prohibiting a protocol, such as a file transfer protocol (ftp) request. If the rules permit the packet to be forwarded, firewall node 1510 may later open (see, e.g., step 3500) another TCP connection to the packet destination address, such as server 1050.

[041] Before opening up a connection or sending a packet to destination server 1050, firewall node 1510 may read the 5-tuple address associated with the packet (step 3350). The 5-tuple address is then processed to determine whether the

5-tuple address is within the region of addresses assigned to firewall node 1510 in 5-tuple space (step 3400). If the 5-tuple address is within the assigned region in 5-tuple space, firewall node 1510 may send the packet with the 5-tuple address to destination server 1050. If the 5-tuple address is not within the assigned region in 5-tuple space, firewall node 1510 may determine another modified 5-tuple address for the packet. The modified 5-tuple address is determined such that the modified address is within a 5-tuple address region assigned to firewall node 1510. Exemplary steps associated with determining whether the 5-tuple address is within the assigned region and determining a modified 5-tuple address are described below with respect to FIG. 6. By way of example, FIG. 5 depicts a 5-tuple address 5310 that is within the region assigned to firewall node two 1510, while 5-tuple address 5320 depicts a 5-tuple that is not within the assigned region of node 1510. Referring again to FIG. 3, firewall node 1510 may send the packet with the modified 5-tuple address to destination server 1050 through second network 1210, with the packet having a 5-tuple address that does not conflict with any of the other firewall nodes 1520-1530 (step 3500-3600).

[042] FIG. 6 depicts exemplary steps for determining an N-tuple address. Client 1010 may attempt to make a TCP connection to server 1050 by sending IP packets to destination server 1050 through first network 1205. Firewall cluster 1500 may then receive the IP packet from client 1010 and select one of the firewall nodes 1510-1530 to further process the packet and the corresponding TCP connection. For example, selected firewall node 1510 may read the 5-tuple address associated with the received IP packet, replace the packet source address (e.g., 4010 at FIG.

4A) with the firewall cluster address (e.g., 4110), and determine whether the packet's 5-tuple address is within the region in 5-tuple space assigned to firewall node 1510. If so, firewall node 1510 may send the packet with the 5-tuple address. If that 5-tuple address is not within the region in 5-tuple space assigned to firewall node 1510, firewall node 1510 may determine a new (or modified) 5-tuple address and then send the packet to its destination.

[043] To determine a 5-tuple address, firewall node 1510 may start with a 5-tuple value and use that 5-tuple value as an initial 5-tuple address (step 6100). Firewall cluster 1510 may determine a quadrant identifier based on a hash function (step 6200), and determine whether the quadrant identifier corresponds to the firewall node number (step 6300). If the quadrant identifier matches the firewall node number, indicating the 5-tuple address is within the node's assigned region in 5-tuple space, the packet is sent with the initial 5-tuple address (steps 6400-6500). If the quadrant identifier does not match the firewall node number, firewall node 1510 may determine a modified 5-tuple address by adjusting one of the 5-tuple address values (step 6600). Firewall node 1510 may send the packet to its destination using the modified 5-tuple address (step 6900). The following provides a more detailed description of steps 6100-6900, with reference to FIGs. 6-8.

[044] Firewall node 1510 may use a 5-tuple value representative of the firewall cluster 1500 address and use that 5-tuple as an initial 5-tuple address (step 6100). For example, firewall node 1510 may read a 5-tuple address depicted in FIG. 4A, i.e., source address 4010, source port 4020, destination address 4030, destination port 4040, and protocol 4050. Firewall node 1510 may replace source

address 4010 with firewall address 4110 and replace source port 4020 with firewall port 4120 (see FIGs. 4A and 4B) and then determine a quadrant identifier for the 5-tuple address based on a function, such as a hash function (step 6200). Generally, a hash function maps something into something else. For example, a hash function may map (or transform) a string of characters or numbers, such as bits or bytes, into a shorter length string or value of bits or bytes. In this case, the hash function maps the 13 bytes of the 5-tuple address to a 1-byte value, which is then modulo divided by the total number of firewall nodes to yield a quadrant identifier.

[045] Although the embodiment above uses the firewall cluster 1500 address as an initial address, any other 5-tuple value may be used instead.

[046] FIG. 7 depicts in block diagram form, a system for determining the quadrant identifier using a hash function and modulo division. Referring to FIG. 7, a 5-tuple having 13 bytes serves as an input to a Hash Function Module 7200. Hash Function Module 7200 may determine a 1-byte hash value. In one embodiment, the Hash Function Module 7200 may sum all of the bytes of the 13-byte 5-tuple address and then truncate the answer to eight bits (or 1-byte). One skilled in the art would understand that any other hash function or transform that maps an N-tuple address to another value may be used instead. The 1-byte output of the Hash Function Module 7200 may further serve as an input to Modulo M divider 7300. Modulo M divider 7300 may function to Modulo M divide the 1-byte output 7110, with M equal to the number of firewall nodes. Returning to the above embodiment depicted in FIG. 1 having three firewall nodes, Output 7120 of the Modulo M divider 7300 is equal to the following:

$$\text{Output} = \text{Remainder}\left(\text{Hash Function}_{output} \div \text{Firewall}_{number}\right)$$

where Output 7120 is the output value of the Modulo M divider 7300; Hash Function_{output} is the output value of Hash Function Module 7200 (e.g., a 1-byte value between 0 and 255), Firewall_{number} represents the number of firewall nodes in firewall cluster 1500 (e.g., a value of 3), and Remainder represents the remainder value of the division operation. In this example, the value of Output 7120 may be one of three possible values 0, 1, or 2 (or, alternatively 1, 2, or 3). The quadrant identifier 7120 may thus have three possible values, namely, zero, one, or two, which represents the three regions assigned to the corresponding three firewalls.

[047] FIG. 8 depicts three quadrants labeled with corresponding quadrant identifier values 8100, 8200, and 8300. As can be seen, the three quadrant identifiers represent one of the three firewall nodes 1510-1530. As such, when a 13-byte 5-tuple address results in a quadrant identifier value of zero, the 5-tuple address is within quadrant zero 8300, which in this example is assigned firewall node zero 1530. Similarly, if a 5-tuple address results in a quadrant identifier of value of two, the 5-tuple address is within quadrant two, which in this example is assigned firewall node two 1510. By using the Hash Function Module 7200 and Modulo M divider 7300, firewall cluster 1500 may determine, based on the quadrant identifier, whether a 5-tuple address is within a firewall node's assigned region in 5-tuple space. For example, if firewall zero 1530 determines that a 5-tuple address has a quadrant identifier of "2" 8200, then the 5-tuple address cannot be used by firewall node zero 1530 (or by node 1520).

[048] One of ordinary skill would recognize that any other approach may be used to determine whether a 5-tuple address is within an assigned region in N-tuple space including, for example, neural networks and Euclidean distance classifiers (e.g., mean squared or least squared error classifiers).

[049] Referring again to FIG. 6, with the quadrant identifier determined, firewall cluster 1500 (or a firewall node therein) may then determine whether the quadrant identifier corresponds to the firewall node (step 6300). As noted above, if the quadrant identifier value is two for a 5-tuple address associated with firewall node two 1510, the quadrant identifier matches the firewall node number, namely, two. As such, the initial 5-tuple address is within the correct region in 5-tuple space assigned to firewall node two 1510. The firewall node, such as firewall node 1510, may then send the packet to its destination server with that initial 5-tuple (steps 6400-6500).

[050] If the quadrant identifier does not match the firewall node number, firewall node 1510 may determine a modified 5-tuple address by adding to one of the 5-tuple address values, the difference between the quadrant identifier and desired quadrant identifier value (i.e., the firewall node number) (step 6600). For example, when the quadrant identifier is one, firewall node 1510 may determine a modified 5-tuple address by first determining the difference between the quadrant identifier (e.g., one) and firewall node number of node 1520 (e.g., two). The difference, in this case equal to a value of two, is then added to any one of the following: source address 4010, source port 4020, destination address 4030, destination port 4040, and protocol 4050. In one embodiment, firewall node 1510

may add the quadrant identifier (e.g., equal to a value of one) to a source port value (e.g., equal to a value of 4096), with the modified 5-tuple having a new source port value of 4097. Although the description herein refers to a packet, such as an IP packet, any datagram or information transformation mechanism may be used instead.

[051] Systems and methods consistent with the present invention may thus use a multidimensional address space for determining an address of a packet, such that addressing conflicts are eliminated. Moreover, the use of the multidimensional address space may enable a firewall cluster to operate in a high traffic network environment with reduced addressing conflicts.

[052] The systems disclosed herein may be embodied in various forms including, for example, a data processing unit, such as a computer that also includes a database. Moreover, the above-noted features and other aspects and principles of the present invention may be implemented in various environments. Such environments and related applications may be specially constructed for performing the various processes and operations of the invention or they may include a general-purpose computer or computing platform selectively activated or reconfigured by code to provide the necessary functionality. The processes disclosed herein are not inherently related to any particular computer or other apparatus, and may be implemented by a suitable combination of hardware, software, and/or firmware. For example, various general-purpose machines may be used with programs written in accordance with teachings of the invention, or it may be more convenient to

construct a specialized apparatus or system to perform the required methods and techniques.

[053] Systems and methods consistent with the present invention also include computer readable media that include program instruction or code for performing various computer-implemented operations based on the methods and processes of the invention. The media and program instructions may be those specially designed and constructed for the purposes of the invention, or they may be of the kind well known and available to those having skill in the computer software arts. Examples of program instructions include, for example, machine code, such as produced by a compiler, and files containing a high level code that can be executed by the computer using an interpreter.